TIN HỌC CƠ SỞ 1 (Sơ lược)

Phần 1. ĐẠI CƯƠNG VỀ TIN HỌC

Chương 1. Thông tin và xử lý thông tin

1.1. Khái niệm về thông tin

- **Thông tin (Information)** là điều mà người ta có thể hiểu biết được. Nó có nhiều dạng: văn bản, hình ảnh, âm thanh, mùi vị, video, … được **Lưu trữ (Save)** trên: nhiều chất liệu như băng từ, đĩa từ, đĩa quang, thẻ nhớ, mạng
- **Tin học (Informatics)** là nghành khoa học nghiên cứu về việc **Xử lý (Process)** thông tin bằng **Máy tính điện tử (MTĐT, Computer, PC.** Mục tiêu của Tin học**:** Phát triển và sử dụng MTĐT để nghiên cứu cấu trúc của thông tin, phương pháp thu thập, lưu trữ, tìm kiếm, biến đổi, truyền tin và ứng dụng vào các lĩnh vực khác nhau của đời sống.
- **CNTT (Information Technology, IT)** là ngành ứng dụng các phương pháp khoa học, các phương tiện và công nghệ kỹ thuật máy tính và viễn thông, nhằm tổ chức khai thác và sử dụng có hiệu quả các nguồn tài nguyên thông tin vô cùng phong phú, tiềm năng và hữu ích trong mọi hoạt động của con người.
- **Đối tượng (Object)** đưa vào xử lý gọi là **Đầu vào (Input)**, kết quả xử lý là **Đầu ra (Output)**.
- Tuy nhiên, vì máy tính chỉ sử dụng điện, quang hay từ, chỉ có 2 **trạng thái** (**bit**): 0 là không có, 1 là có [điện, quang, từ]. Mọi vấn đề cốt lõi, phức tạp, rắc rối đều dự trên đặc điểm này!
- Người ta cần phải **Mã hóa (Encoding)** thông tin thành dạng mà máy tính tiếp thu và xử lý được ở dạng bit. Thông tin được mã hóa thành **Dữ liệu (Data)**. Máy tính chỉ xử lý trực tiếp được dữ liệu!
- Như vậy quá trình con người làm việc với máy tính là:

Thông tin → Dữ liệu → Xử lý → Dữ liệu → Thông tin.

- Biến đổi từ Dữ liệu thành Thông tin gọi là Giải mã (Decoding).
- Dữ liệu cũng là một dạng thông tin. Thông tin không phải là Dữ liệu trực tiếp!
- Độ lớn của thông tin T đo bằng $\log_2$(<mã của T>) là số các bít mã của T.

Ví dụ:

Cộng 2 với 3, người xử lý cho kết quả là 5. Nhưng với máy, khi ta gõ số 2 từ bàn phím thì vào máy phải được mã hóa thành 00000001 và 3 thành 00000011, máy cộng lại thành 00000101 rồi giải mã ra màn hình là 5.

1.2. Mã hóa và giải mã

- Đơn vị thông tin nhỏ nhất cho máy tính xử lý được là hai bít **0 và 1.**
- Ngày xưa mã hóa bằng máy đục lỗ trên các tấm bìa đục lỗ, hay băng đục lỗ, mã hóa hay giải mã nhờ bóng đèn điện tử, hoặc đèn điện và tế bào quang điện.
- Ngày nay đưa dữ liệu vào máy từ: bàn phím, thiết bị vào/ra hoặc từ mạng Internet/không dây, nhờ tia laser với tế bào quang điện và chip điện tử, đưa dữ liệu ra màn hình chuyển các dãy bít thành các **Điểm ảnh (Pixel)**.
- Dùng nhóm **n** bit**,** ta có thể mã hóa được $\mathbf{2^n}$ trạng thái của một loại Thông tin**.**
- Ban đầu, ta tạm dùng bộ 256 kí tự gọi là bộ mã **ASCII (American Standard Codes for Information Interchange).** Bảng chữ cái theo tiếng Anh!
- Kí tự **(Character)** là các chữ cái, chữ số, các kí hiệu cơ bản và **Kí tự điều khiển** (không hiện ra màn hình mà chỉ để điều hành công việc của máy tính).
- Vì $256 = 2^8$, nên người ta dùng các nhóm 8 bít để có thể mã/giải mã cho thuận tiện. Ta nói 8 bit là một **Từ máy.** Máy xử lý được các dãy bít chia hết cho 8**.**
- Số thứ tự các kí tự ở hệ đếm **DEC (DECimal, thập phân)** gọi là **Mã thập phân của kí tự** đó, từ 0 đến 255, hay dạng bit là từ 00000000 đến 11111111.

Ví dụ:
'A' = 65, 'B' = 66,…. 'a' = 97, 'b' = 98,…. '0' = 48, Esc = 27. Dấu cách = 32, ↓ = 10, ← = 13, ↵ = (10,13). BackSpace = 8, Table = 9, bình phương = 251.

- Khi gõ một phím thì kí tự được chuyển thành dãy bít, truyền vào máy nhờ **Đường truyền (Bus)** vào các **Cổng song song (Parallel Port)** hoặc **Cổng nối tiếp (Serial Port)**.
- Hiện một ký tự ở máy ra màn hình tức là chuyển chùm **8 bit** thành **hình ảnh** theo **Mẫu (Font)** xác định nào đó, đi theo **Cable màn hình.** Mỗi mẫu có thể nhiều bít hơn! Như vậy, mỗi kí tự trước hết có Mã và có Mẫu của nó!
- Các đường tuyền từng chùm bit gọi là **Đường truyền song song (Parallel Bus)**. Trái lại nếu đi từng bit một là **Đường truyền nối tiếp (Serial Bus)**.
- Ngày xưa người ta có một cách mã hóa/giải dạng **Tạnh-Tè (0-1)** để truyền tin qua sóng radio, theo đường truyền nối tiếp!
- Nay nhờ **Modem Dial-up** hoặc **ADSL (Asymmetrical Digital Subscriber Line** (Đường thuê bao kỹ thuật số bất đối xứng: tải xuống nhanh hơn).
- Modem Dial-up có tốc độ truy nhập mạng tối đa là **56 kbps (56.000 bits per second**).
- Modem ADSL truyền trực tiếp bằng kỹ thuật số có tốc độ tối đa là **100 Mbps**.
- Thuật ngữ **Modem** là ghép nối giữa **Modulation** (Chuyển tín hiệu Analog thành Digital) và **Demodulation** (Chuyển Digital thành Analog).

- **Kỹ thuật tương tự (Analog)** là kỹ thuật mà thông tin ở đầu ra thu được càng giống càng tốt so với thông tin ở đầu vào. Do đó, với kỹ thuật tương tự mới có các thuật ngữ: **HQ (Chất lượng cao)**, **Hi-Fi (High fidelity), VHS (Very hight sound), …**
- **Kỹ thuật số (Digital)** là kỹ thuật mã hóa/giải mã thông tin hoàn toàn chính xác tới từng bit một (hết sức chính xác khi đường truyền được thông suốt).

1.3. Đơn vị thông tin

- Đơn vị **bit** là nhỏ nhất, viết tắt là **b** (chữ thường).
- Nhóm 8 bit gọi tắt là 1 **Byte**, viết tắt là **B** (chữ hoa), tương đương một kí tự nếu dùng ASCII, về mặt độ lớn thông tin.
- Ngày nay có **Bộ kí tự Quốc tế** (**Unicode**) gồm 65536 kí tự ($65536 = 2^{16}$).
- **Unicode** phải dùng 2 Bytes để mã hóa 1 kí tự.
- Các đơn vị dẫn suất khác của **B** để chỉ độ lớn của thông tin:

1 KB (Ki lô Byte)	=	1024 Bytes	≈ 1 trang sách. ($1024 = 2^{10}$)
1 MB (Mega Byte)	=	1024 KB	≈ 1.000 trang sách.
1 GB (Giga Byte)	=	1024 MB	≈ 1.000.000 trang sách.
1 TB (Têta Byte)	=	1024 GB	≈ 1.000.000.000 trang sách.

- Gói dữ liệu <u>được đặt tên</u> gợi nhớ và <u>lưu trên thiết bị lưu trữ</u> gọi là một **File** (**Tệp**, **Tập tin).**

Một [vài] tệp lại có thể gom lại thành một **Thư mục (Directory/Folder)** theo một ý tưởng nào đó của **Người sử dụng (User)** và đặt cho nhóm đó một cái tên gợi nhớ. Một [vài] thư mục (và [vài] tệp) lại có thể gom lại thành một thư mục lớn hơn gọi là Thư mục Mẹ của chúng. Như vậy, một thư mục chứa [các] **Thư mục con (Sub-directory**, **Sub-folder)** và có thể một số tệp lẻ tẻ nữa. **Thư mục không chứa gì cả** gọi là **Thư mục rỗng (Empty Directory).** Các thư mục lại nằm trên các **Thiết bị lưu trữ (Storage Devices): Đĩa cứng (Hard Disk, HD)**, **Đĩa mềm (Floppy Disk, FD)**, **Đĩa quang (Compact Disk)**, **Thẻ nhớ (Memstick),** hay trên **Đám mây trên mạng (Cloud Computing).** Thư mục trực thuộc một ổ đĩa gọi là **Thư mục gốc (Root-Folder). Ổ đĩa (Drive)** là một phân vùng của một đĩa, có bộ phận thể đọc dữ liệu ra và/hoặc có thể viết dữ liệu vào.

- Các Files dữ liệu của MTĐT tạo thành **Tài nguyên (Fortune)** của MTĐT đó.
- Ngoài các kiến thức trên, sinh viên còn phải hiểu sâu sắc về các Nguyên lý hoạt động của MTĐT và Cơ sở Toán học và Logic của Tin học nữa!

Sinh viên hay bất cứ ai muốn tìm hiểu điều gì, đều có thể gõ từ khóa vào trang Google để tìm. Nên tìm hiểu kỹ về cách gõ từ khóa, chi tiết xem trên Google với "8 thủ thuật tìm kiếm trên Google"**.** Nhớ gõ cả dấu nháy kép!!!

Chương 2. Đại cương về máy tính điện tử

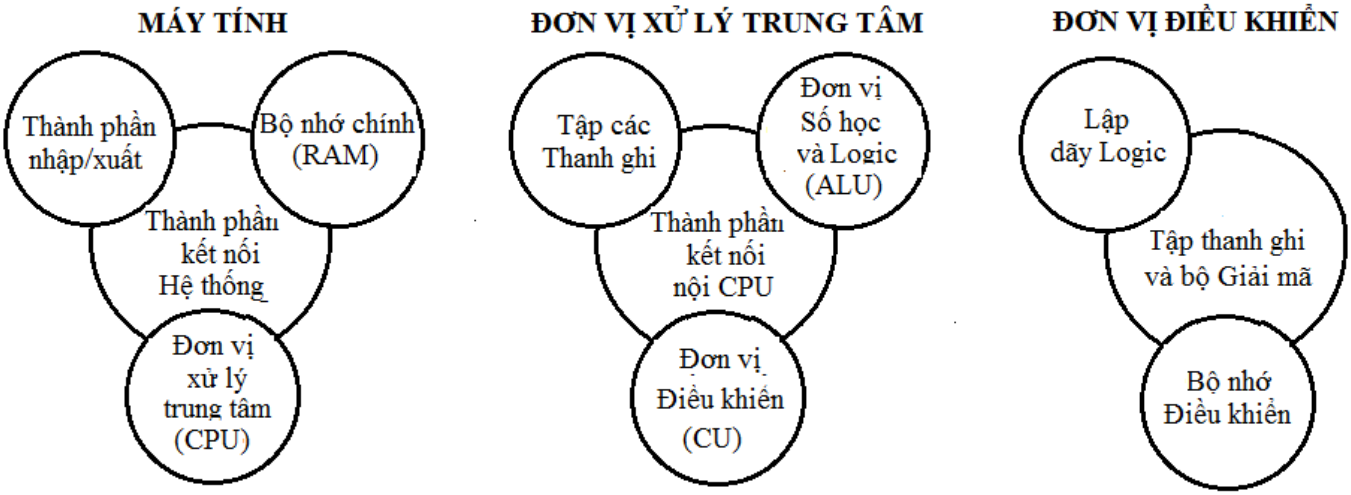
2.1. Kiến trúc MTĐT

2.1.1 Chức năng của MTĐT (5 chức năng chính)

- Nhập dữ liệu, Lưu trữ dữ liệu, Xử lý dữ liệu, Xuất dữ liệu, và Điều khiển các thiết bị ngoại vi.

2.1.2 Cấu trúc chung

MTĐT bao gồm các bộ phận chính ghép nối với nhau như sơ đồ sau:



- **Bộ xử lý trung tâm (Central Processing Unit, CPU), Bộ nhớ (Memory, MEM), Bộ điều khiển (Keyboard, Mouse), Thiết bị vào/ra (Input/Output Devices)**...

1. Bộ xử lý trung tâm (CPU = Central Processing Unit)

- Là thiết bị chính của MTĐT, phân tích và thực hiện các lệnh do chương trình đưa từ bộ nhớ, ví như bộ óc của con người. Trong CPU có **Đơn vị Điều khiển (Control Unit, CU)** để nhận lệnh, giải mã lệnh và điều hành chung, **Đơn vị Số học và Logic (Arithmetic-Logic Unit, ALU)** thực hiện các phép toán số học và logic, **Bộ nhớ đệm (Cache),** các **Thanh ghi (Registers).**
- Registers thường trực trong CU, nơi đón các loại dữ liệu phục vụ cho việc thực hiện lệnh như Mã lệnh, Địa chỉ dữ liệu và Dữ liệu để đưa vào cho CPU xử lý, và Địa chỉ đưa kết quả ra sau khi CPU xử lý rồi điều khiển giải mã đưa ra ngoài. Registers ví như phòng thường trực của CPU.
- Giá trị của CPU chủ yếu phụ thuộc vào **Tốc độ (Speed)** xử lý, **đo bằng Hertz (Hz).** Hertz là số xung nhịp trong 1 giây. Số xung nhịp dùng để đồng bộ mọi hoạt động của MTĐT.
- CPU nhanh/chậm còn phụ thuộc yếu tố **Kiến trúc (Structure)** ở các thanh ghi:

16 bit, 32 bit và 64 bit, (được thiết lập khi cài đặt Hệ điều hành). Các con số đó nói lên số bit của mỗi thanh ghi. Số đó càng lớn thì càng nhanh. Chẳng hạn với kiến trúc 16 bit thì chỉ làm việc được với số ô nhớ tối đa là 2^{16}, địa chỉ ô nhớ nhiều hơn nữa thì máy tính không thể với tới được. Việc xử lý các phép tính số nguyên cũng không tính toán trực tiếp được các số > 65535, mà phải chia làm nhiều đợt tính toán và ghép kết quả lại nên lâu hơn. Hiện đại nhất bây giờ là kiến trúc 64 bit: truy cập được nhiều ô nhớ, tính được số lớn hơn và chạy được nhiều lệnh hơn. Tuy vậy, nó cũng gây ra một số phiền toái nhất định!

- Ngoài ra yếu tố **tốc độ đường truyền (BUS speed, cũng đo bằng MHz, hoặc MB/s)** cũng góp phần cho tốc độ của MTĐT. Tốc độ BUS càng cao thì nhiều dữ liệu có thể truyền ra/vào CPU trong một đơn vị thời gian nhanh hơn. Muốn máy chạy nhanh thì nên mua máy đồng bộ!
- CPU có thể có đa nhân từ năm 2005 như lõi kép, lõi tứ. Nhờ đó, máy tính có thể xử lý nhiều nhiệm vụ trong một thời điểm!
- Tất nhiên là ta còn phải lưu tâm đến thương hiệu, nhà sản xuất: **Pentium M, Pentium II, …, IV, Celeron, Intel Dual Core, Intel Core 2 Duo, Intel Core 2 Quad, Core i2, …, i7, Intel Xeon**….

Ví dụ:

- Pentium IV có tốc độ 2.4 GHz, tức là trong 1 giây có thể thực hiện được $2.4*2^{30}$ phép toán, tức là gần 2,4 tỷ thao tác cơ bản (Xung nhịp)/1 giây. Hiện nay, **Core i7** đang được đánh giá là nhanh nhất.

2. Bộ nhớ (Memory)

- Là dãy các ô nhớ cỡ 1 Byte. **Chức năng của Memory** là lưu trữ dãy lệnh của chương trình, dữ liệu ngay trước và sau khi xử lý.
- Các ô nhớ được đánh số từ 0 đến Max… (hết khả năng của nó). Số thứ tự các ô nhớ gọi là **Địa chỉ vật lý.** Để dễ truy nhập người ta đánh số các ô nhớ theo Địa chỉ logic: **Đoạn và Độ lệch [segment,ofset]**, (giống như phố và số nhà).
- Hiệu suất của bộ nhớ phụ thuộc vào **Dung lượng (Volume)** đo bằng Byte), **Tốc độ (Speed),** đo bằng Hz hoặc MB/s. Tốc độ lưu vào có thể khác tốc độ ra!
- **Bộ nhớ gồm: Bộ nhớ chính (Main Memory)** và **Bộ nhớ phụ hay ngoài (Auxiliary Memory)**,

a)- Bộ nhớ chính (Main Memory) hay còn gọi là Bộ nhớ trong:

- Bộ nhớ trong có dung lượng nhỏ nhưng tốc độ truy cập rất nhanh, nó gồm: **ROM và RAM.**

- **ROM (Read-Only Memory)** là bộ nhớ chỉ đọc ra được, mà không ai sửa chữa hay xóa được các thông tin trong đó, trừ nhà sản xuất chế tác lại,
- ROM ví như bia đá, hay cuốn sách. Khi mất điện dữ liệu của nó vẫn còn.
- **RAM (Random Access Memory)** là bộ nhớ truy cập ngẫu nhiên, có thể đưa dữ liệu vào hay lấy dữ liệu ra từ bất cứ địa chỉ nào khả dụng.
- Băng từ là một ví dụ Bộ nhớ Truy cập tuần tự **(Succesif** hay **Sequential Memory Device)**, trái ngược với **Ngẫu nhiên (Random).**
- Dữ liệu trên RAM chỉ là tạm thời, sẽ mất dữ liệu đi khi mất điện. Trên ROM thì còn mãi mãi!
- **MTĐT** từ khi khởi động, càng chạy nhiều chương trình thì RAM cũng cạn dần và máy chạy chậm đi. Thỉnh thoảng ta thường phải **Làm tươi mát lại** (**Refresh**) bộ nhớ hoặc khởi động lại máy để máy chạy nhanh như ban đầu!
- **MTĐT** có RAM với **Dung lượng (Volume)** càng lớn và **Tốc độ (Speed**) càng nhanh thì giúp máy chạy càng nhanh!

Ví dụ: Với tốc độ 800 MHz và 2 MB/s, tức khoảng 800 triệu đợt ghi vào hay lấy ra, mỗi đợt chuyển được 2 MB dữ liệu 1 giây. Càng ngày, người ta càng làm được những thanh ram rất nhỏ, dung lượng lớn và tốc độ rất nhanh!

Chú ý:
- Tuy nhiên MTĐT nào mà RAM có dung lượng ít thì ta thường tạo ra **Bộ nhớ ảo (Virtual Memory)** bằng cách lấy một phần ổ đĩa cứng làm RAM, nhưng không thể chuẩn bằng RAM thật.
- Mặt khác, máy có RAM nhưng không có ổ cứng, thì người ta lại có thể dùng một phần RAM làm ổ cứng. Ổ đĩa được tạo ra đó gọi là **Ổ đĩa ảo (Virtual Drive).** Một số quán Internet dùng máy tính không cần đến ở đĩa cứng mà dùng RAM làm ổ đĩa ảo, không sợ nhiễm virus, vì khởi động lại máy, virus mất hết.

Máy tính tuần tự thực hiện các lệnh từ chương trình mà USER yêu cầu:

- CU làm việc theo **Chu trình**:

Nhặt lệnh từ **Hàng đợi lệnh (Queue) >** Giải mã lệnh, xem cụ thể phải làm gì theo các **Tham số/Đối số (Parameters)** nào > Đặt lệnh vào **Thanh ghi Lệnh >** Đặt tham số vào **Thanh ghi Địa chỉ >** Đặt dữ liệu vào **Thanh ghi Dữ liệu** theo **Địa chỉ (Address)** của nó từ RAM. Rồi ALU tính toán, xong thì gửi kết quả vào **Thanh ghi Tích lũy** để đưa ra **Bộ nhớ (RAM) trả về cho chương trình.**
Cũng ví như ở điện thoại:
Gõ *100*<mã thẻ># gọi thì Điện thoại làm gì?

Ví dụ:
Việc cộng a + b = c. CPU thực hiện lệnh (Xin cấp phát bộ nhớ cho a, b và c). Sau đó CU nhận lệnh (Nhập dữ liệu) thì CU chuyển dữ liệu từ bàn phím vào RAM ở các ô nhớ a, b. Lệnh tiếp (Lấy dữ liệu ở ô nhớ a cộng dữ liệu ở ô nhớ b ở ALU, đưa kết quả vào ô nhớ c), Lệnh tiếp (Xuất ra màn hình) thì nó giải mã c và xuất ra màn hình đúng chỗ theo yêu cầu lệnh ...

b)- Bộ nhớ ngoài (External Memory) hay còn gọi là Thiết bị lưu trữ (Storage Devices):

- Bộ nhớ ngoài gồm các thiết bị lưu trữ như các đĩa, thẻ nhớ, băng từ, dung lượng lớn nhưng tốc độ truy cập chậm hơn bộ nhớ trong.
- **Đĩa cứng HD (Hard Disk),** có thể tới: **80 GB, 160 GB, 250 GB, 320 GB, 500GB, ..., 1 TB**.
- **Đĩa cứng cổ**: Nó phải quay tít liên tục để đầu đọc quét các dữ liệu.
- **Đĩa hiện đại** dùng công nghệ đặc biệt như một thẻ nhớ, không tốn điện, cắm vào cổng USB, không cần quay tít…
- Một đĩa cứng vật lý nên chia nhỏ thành nhiều **Phân vùng (Partition),** có bộ phận đọc/ghi riêng tạo thành một **ổ đĩa HDD (Hard Disk Drive).** Thiết bị lưu trữ nào có đầu đọc/ghi đều gọi **là thiết bị vào/ra (I/O Device)**, nghĩa là điều khiển đưa dữ liệu vào MTĐT, và nhận dữ liệu từ MTĐT ra. Có thể có thiết bị chỉ có vào hoặc chỉ có ra!
- **Đĩa quang (Compact Disk, CD).** Compact là kín không để ánh sáng lọt vào. Trên mặt đĩa có các hạt phản quang hay không phản quang tương ứng với các bít 1 hay 0**. Đọc/Ghi CD bằng tia Laser:** Khi tia laser chiếu vào một điểm trên đĩa nó phản xạ sang tế bào quang điện để nhận lấy kết quả là 1 hoặc 0 (sáng hay tối).
- Đĩa loại **CD có dung lượng 740 MB.**
- Có nhiều loại **CD** như: **VCD (Video Compact Disk), CD-R (Recordable)** dùng để ghi vào được để trở thành một **CD-ROM**, hay **CD-RW (Rewritable)** cũng để ghi vào – xóa đi – ghi lại rất nhiều lần, gần 500 lần.
- Đĩa quang video **DVD (Digital Video Disk)** kỹ thuật cao hơn, dung lượng lên tới **4.7 GB** (hơn 6 lần đĩa CD). Tương tự như CD, cũng Có nhiều loại **DVD** như: **DVD-R** dùng để ghi vào, **DVD-RW**.
- **Thẻ nhớ (Memstick**) rất hay dùng cho máy ảnh hay quay phim, iPhone, iPad, **[Ổ đĩa] USB-Flash** cắm vào cổng USB, có phiên bản và dung lượng ngày càng được cải tiến. Chúng được ứng xử bình đẳng như các Ổ đĩa.
- Toàn bộ dữ liệu trên ổ cứng tạo thành **Tài nguyên (Fortune)** của một MTĐT.
- Computer liên hệ với tất cả các thiết bị vào ra qua các cổng giao tiếp của nó.
- Độ bền của đĩa cứng cũng rất lâu. Đĩa CD, DVD độ bền cũng chỉ được từ 5-10

năm thôi. Nhiều USER tận dụng mạng Internet để dữ liệu quan trọng trên mạng (Google Drive, DropBox, Gmail,…), trừ những dữ liệu bí mật chưa mã hóa…
- Tốc độ truy cập tốt lần lượt giảm dần là từ CPU, Cache, Registers, ROM, RAM, HD, CD, USB đến Memstick.
- Ngày nay, có mạng Internet, việc dùng đĩa CD hay DVD đang ít dần!

Chú ý:
Cần phân biệt:
- **Đĩa ≠ Ổ đĩa.** Đĩa là thiết bị lưu trữ, Ổ đĩa là thiết bị cả lưu trữ và điều khiển vào/ra/.
- **Drive** là Ổ đĩa ≠ **Driver** là trình điều khiển thiết bị.
- **Driver là (chương) trình điều khiển thiết bị** ngoài hay còn gọi là **Ngoại vi (Peripheral)** khi gắn vào máy qua một **Cổng giao tiếp (Port)** nào đó. Driver giúp cho Hệ điều hành biết cách sử dụng thiết bị đó. Nhà sản xuất thiết bị cũng thường cải tiến và cho phép cập nhật miễn phí Driver từ trang web của họ.

Ví dụ:
Khi mới mua một cái lò vi sóng (như một thiết bị mới) về, cả nhà không ai biết cách dùng. Khi đó sách hướng dẫn sử dụng (giống như trình điều khiển) được đọc cho mọi người nghe hiểu (giống như cài driver). Từ đó trở đi mọi người trong gia đình (giống như Hệ điều hành) biết cách dùng!

3. Bộ điều khiển

a)- Bàn phím (Keyboard)
- Bàn phím có thể có loại cho **máy bàn nằm (Desktop), máy đứng tháp (Tower)** hay cho **máy xách tay (Laptop, Notebook).** Bàn phím có các phím để gõ kí tự từ tệp **Bộ kí tự trên nó (Console**), kể cả các phím điều khiển.
- **Bàn phím** vừa là **Thiết bị vào chuẩn** (vì nó đưa các kí tự vào máy) vừa là một **Thiết bị điều khiển.**

Cách gõ bàn phím:

Phím chữ cái:
- **Chế độ chữ thường** (**đèn Caps Lock tắt**): Gõ cho ra chữ thường, với Shiftcho ra chữ HOA. **Chế độ chữ hoa (đèn Caps Lock sáng)**: Gõ cho ra chữ HOA, với Shift cho ra chữ thường. Muốn gõ một đoạn dài nhiều chữ in HOA ta bật sẵn đèn Caps Lock. Phím Caps Lock dùng để chuyển đổi hai chế độ đó.
- Phím có 2 kí tự khác: Gõ binh thường cho kí tự dưới, với Shift cho kí tự trên.

Phím điều khiển: Chỉ có nhiệm vụ điều khiển:

- Các mũi tên dịch chuyển con trỏ văn bản lên, xuống, sang trái, sang phải và xuống dòng dưới (↑, ↓, ←, → và ↵ tương ứng). Để ↓ có hiệu lực thì trước đó đã gõ phím ↵ một lần nào đó rồi.
- Các phím F1, …, F12 liên quan đến điều khiển Hệ điều hành.

Ví dụ:

F1, ..., F12	các phím chức năng… Tìm hiểu thêm!
F1	Thường hiện ra việc trợ giúp của phần mềm đang chạy.
F12	Thường dùng khi chạy cài đặt hệ điều hành...
F4	Thường để kết thúc phần mềm đang chạy.
Esc (escape)	để lùi về bước trước.
BackSpace	để lùi xóa kí tự bên trái vạch nhấp nháy (nếu có), hoặc nhẩy lên dòng trên.
Tab	để đẩy phần dòng bên phải con trỏ sang phải một đoạn.
PrintScreen	để chụp ảnh màn hnh: Chạy Paint, chọn Paste để dán vào, cắt, sửa [và lưu nếu muốn].
Pause/Break	để tạm dừng hoạt động của chương trnh đang mải mê chạy.
Insert	để đổi chế độ chèn/đè khi gõ văn bản tại vạch nhấp nháy.
Home	để vạch nhấp nháy lập tức về đầu dòng hiện thời.
End	để vạch nhấp nháy lập tức về cuối dòng hiện thời.
PageUp	để chuyển nhanh lên trang trước (nếu có).
PageDown	để chuyển nhanh đến trang sau (nếu có).
Delete	để xóa kí tự đầu bên phải vạch nhấp nháy, kéo phần dòng bên phải sang trái. Cũng để đưa Đối tượng vào Sọt rác!
Enter	để khẳng định nhập dữ liệu hoặc đưa phần xuống dòng dưới. Con trỏ chỉ xuống dòng được khi đã dùng Enter một khi nào đó.
CapsLock	để chuyển đổi chế độ HOA và thường. Đèn Caps Lock = Chế độ chữ HOA. Khi gõ đoạn toàn chữ hoa thì nên bật bật Caps Lock.
NumLock	để bật/tắt phần phím số phụ tiện cho việc nhập dữ liệu số. Khi bật (Đèn sáng) thì NumLock có hiệu lực! Ở nhiều Laptop phần phím số phụ này lộn vào giữa, khó phát hiện ra.
Shift,Ctrl,Alt	để hỗ trợ các phím khác mới có nghĩa. Tự nó không có nghĩa! Ta có thể gõ một kí tự theo mẫu: Giữ Alt gõ <mã thập phân> ở phần bàn phím số phụ, ví dụ để gõ x^2 ta có thể gõ x, rồi một tay giữ phím Alt, tay kia gõ 253 là được. Việc này dễ dàng làm được trên máy bàn.

Phần chính của bàn phím:

Esc	F1	F2	F3	F4		F5	F6	F7	F8		F9	F10	F11	F12
~ `	! 1	@ 2	# 3	$ 4	% 5	^ 6	& 7	* 8	(9	) 0	— -	+ =	| \	Back Space
Tab	←→|	Q	W	E	R	T	Y	U	I	O	P	{ [	}]	
CapsLock		A	S	D	F	G	H	J	K	L	: ;	" '	Enter	
□ Shift			Z	X	C	V	B	N	M	< ,	> .	? /	□ Shift	
Ctrl	□	Alt	Dấu cách								Alt	□	Ctrl	

- Phím **F** và **J** có gai giúp người ta tập đánh 10 ngón biết vị trí mốc của ngón tay trái và ngón tay phải!
- Khi gõ một phím thì kí tự ở Console được mã hóa và gửi dãy bít này vào RAM của máy qua các cổng của bàn phím ở **Bo mạch chủ (Main Board).** Trên Bo mạch chủ trước đây có các cổng cổ điển, mỗi thiết bị có một cổng riêng. Đó là cổng **Song song (Parallel)**: Bàn phím cổng mầu tím, **Con chuột (Mouse)** cổng mầu xanh, **Máy in (Printer)** cổng hình thang cân, … Ngày nay, phổ biến là các cổng hiện đại như **Cổng Nối tiếp Tổng hợp (Universal Serial Bus)**, thiết bị nào có đầu cắm vừa vào, đều được.

b)- Chuột (Mouse)

- Chuột (Mouse) là một **Thiết bị điều khiển**, không phải là thiết bị vào/ra
- Chức năng của chuột là vị trí xác định trên màn hình bằng hình ảnh của mũi tên chuột theo các thao tác (nhấp đơn, đúp, nhấp phải hay kéo thả, hay vê phím giữa) để máy xử lý.

Nguyên tắc sau:

- **Nhấp trái (Left-Click)** để **chọn** đối tượng mà nó đang chỉ vào.
- **Nhấp phải (Right-Click)** để **hiện thực đơn** (menu, bảng chọn) áp dụng cho đối tượng.
- **Nhấp đúp (Double-Click)** để **kích hoạt** (**mở** thư mục/chương trnh).
- **Kéo thả (Drag-and-Drop)** để **di chuyển đối tượng** trông thấy.
- **Vê phím giữa** để chuyển các đối tượng lên trên/xuống nhanh.

4. Các thiết bị vào/ra khác (Input/Output Devices)

a)- Màn hình (Screen)

Màn hình **là thiết bị ra chuẩn.** Cấu tạo của màn hình gồm các **Điểm ảnh (Pixel),** mỗi điểm mang theo dữ liệu thể hiện **thông tin về mầu sắc.**

Ví dụ:
Một điểm mầu đỏ chất lượng kém (8 bít) mang dữ liệu là 00001100 mầu số 12, tức là có tất cả 2^8 hay 256 mầu. **Chất lượng mầu (Color Quality)** tốt hơn sẽ là 16 bit, 24 bit hay 32 bit tùy theo cấu trúc phần cứng và thiết lập của người sử dụng (User).

- Mỗi màn hình sau khi thiết lập thì các điểm ảnh được tổ chức thành bảng có số cột và số hàng nhất định, gọi là **Độ phân giải (Screen Resolution)** của màn hình. Chẳng hạn 1024 by 768 lines, tức là 1024 cột và 768 hàng.
- Độ sáng tại mỗi điểm cũng ảnh hưởng đến độ bền của màn hình. Ta thường dùng trình **Lưu dưỡng màn hình (Screen Saver)** giúp màn hình được bền.
- Một số MTĐT có thể có phần mềm tăng giảm được độ sáng chói của màn hình.
- Chất liệu màn hình có thể là Tinh thể lỏng **(Liquid Crystal Display, LCD**) hay Điốt phát quang **(Light Emitting Diot, LED**).

b)- Máy in (Printer) có 3 loại chính:

- **In kim** (thô)**, Phun mực** (máy rẻ, mực đắt) và **Laser** (rất cao cấp và đắt tiền hơn, nhất là laser mầu, nên ít thông dụng cho lớp người dùng bình dân). Kích cỡ máy in cũng có nhiều loại, có cả loại mini in ảnh chuyên dụng. Chức năng cũng có thể tích hợp tem ORC (Chuyển chữ trong ảnh thành chữ văn bản), hay kèm theo Quét ảnh (Scanner).

c)- Nhiều loại thiết bị khác:

Loa (Speaker, Headphones),**Webcam, Micro (**Có loại đi kèm với Headphones), **Scanner, Tivi-Box, ...**

Chú ý:
Muốn có một máy tính tốt cần tham khảo ý kiến các chuyên gia:

- Bộ xử lý trung tâm CPU nhanh, Cache lớn, RAM dung lượng lớn với tốc độ truy cập nhanh, Đĩa cứng HD dung lượng lớn, và phân ra làm nhiều Partitions C: (tối thiểu 50GB), D:, E:, F:,…Không để dữ liệu riêng tư như **Tài liệu [của tôi] ([My] Documents), Music, Pictures,Videos** (do mình tạo ra) ở ổ C: mà để

sang các ổ khác! Màn hình độ phân giải cao cùng chất lượng mầu lớn (tối thiểu 16 bít). Nên mua màn hình LCD hoặc LED. Chọn thương hiệu có uy tín và mọi thứ nên đồng bộ để phát huy hết khả năng của các thiết bị. Nếu mua ổ quang thì mua máy có ổ ghi DVD-RW để tiện làm các đĩa quang lưu trữ hoặc máy có cổng USB và cổng Giao diện đa phương tiện phân giải cao (**High-Definition Multimedia Interface, HDMI**), có khả năng truyền tải video và âm thanh

- **2.2. Nguyên lý Von Neumann**

- **John Von Neumann (28/12/1903 – 8/2/1957)** là một nhà toán học người Mỹ gốc Hungary. Tên khai sinh là Neumann János Lajos.
- Ông cũng là một nhà bác học thông thạo nhiều lĩnh vực đã đóng góp vào vật lý lượng tử, giải tích hàm, lý thuyết tập hợp, kinh tế, khoa học máy tính, giải tích số, động lực học chất lỏng, thống kê và nhiều lĩnh vực toán học khác.
- Năm 1946 lần đầu tiên được John Von Neumann mô tả cấu trúc như đã trình bày ở mục trên.
- Ông ta cũng đưa ra nguyên lý hoạt động cũng gọi là **nguyên lý Von Neumann** như sau:

  1.- MTĐT phải được **điều khiển bằng chương trnh** đã được lưu trữ ở bộ nhớ, theo một kịch bản mà con người ta lập sẵn cho nó.

  2.- MTĐT **truy cập dữ liệu thông qua địa chỉ**. Dữ liệu ở đây gồm dữ liệu vào, ra, trung gian hay các mã lệnh của chương trnh.
- Nguyên lý này đảm bảo tính mềm dẻo tổng quát trong việc xử lý thông tin, người lập trình viết yêu cầu một cách tổng quát đến các địa chỉ của dữ liệu.
- Nguyên lý và cấu trúc này của Von Neumann là một cuộc cách mạng làm tăng tốc độ tính toán rất nhiều, vì trước kia máy chỉ nhận được các lệnh từ băng giấy hoặc bìa đục lỗ và nạp thủ công dữ liệu cụ thể bằng tay, mất rất nhiều thời gian, nhất là khi gặp bài toán cần lặp đi lặp lại nhiều lần.

Các thế hệ của MTĐT, các loại máy tính hiện nay

Thế hệ 1 (1950-1958): Sử dụng bóng đèn điện tử, mạch riêng rẽ, vào số liệu bằng phiếu đục lỗ, điều khiển bằng tay. Máy có kích thước rất lớn, tiêu tốn điện, Tốc độ xử lý rất chậm, khoảng 300-3000 Hz. Điển hình ở thế hệ 1 là ENIAC, EDVAC (Mỹ) hay BESEM (Nga).
- Thế hệ 2 (1958-1964): Xử lý bằng đèn bán dẫn, mạch in. Máy đã có chương trình dịch theo ngôn ngữ Cobol, Fortran và hệ điều hành đơn giản, kích thước lớn, tốc độ từ 10.000- 100.000 Hz, như IBM-1070 (Mỹ) hay MINSK (Nga).
- Thế hệ 3 (1965-1974): Có các vi mạch điện tử cỡ nhỏ, tốc độ khoảng từ

100.000-1.000.000 Hz. Đã có hệ điều hành đa nhiệm, nhiều người cùng sử dụng, có thể in trực tiếp ra máy in. Điển hình như loại IBM-360 (Mỹ) hay EC (Nga).

- Thế hệ 4 (1974-1990): Có các vi mạch đa xử lý với tốc độ từ hàng chục triệu đến hàng tỷ Hz. Đã có 2 loại: MTĐT để bàn và xách tay. Đã có các máy tính chuyên nghiệp. Đã hình thành mạng máy tính (Computer Network). Đã có nhiều ứng dụng đa phương tiện.
- Thế hệ 5 (1990-nay): Chế tạo máy tính mô phỏng theo não bộ của con người, có trí khôn nhân tạo có khả năng tự suy diễn, phát triển các tình huống và hệ thống quản lý kiến thức cơ bản để giải quyết các bài toán đa dạng. Có nhiều loại máy tính: **Máy vi tính (Micro-Computer)**, **Máy tính tầm trung (Mini-Computer)**, **Máy tính lớn (Mainframe-Computer)**, **Máy tính để bàn nằm (Desktop-Computer)**, **đứng (Tower-Computer), Máy chủ (Server-Computer)**, ... Ta còn có **Máy tính nhúng (Embedded-Computer)** là loại được thiết kế chuyên dụng đặt trong các thiết bị điều khiển khác nhau.

2.3. Cơ sở số học và logic của MTĐT

2.3.1. Hệ đếm trong MTĐT

a)- Khái niệm về vài hệ đếm quan trọng

- **Hệ thập phân (DECimal)** gồm 10 chữ số 0,1,…,9. Đếm và các phép tính thực hiện bình thường.
- Khai triển một số tự nhiên D có n chữ số là một Đa thức $P_n(10)$, bậc n với các hệ số lần lượt là các số từ 0 đến 9.

- **Hệ nhị phân (BINary)** gồm 2 chữ số 0 và 1. Đếm 0, 1, 10, 11, 100, 101, 110, 111, 1000,… Công thức: $\mathbf{(10_2)^n = 2^n}$ ở hệ DEC**.**
- Để ám chỉ hệ nhị phân ta viết chỉ số dưới là 2 (trong Toán học) hoặc b ở cuối (trong Assembly).Chẳng hạn 10001110101_2 hay 10001110101b cũng như nhau.
- Đổi một số tự nhiên D từ hệ BIN ra số hệ DEC nhờ khai triển thành đa thức $P_n(2)$, chẳng hạn: $D = 1101_2 = 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = 13$ (ở hệ DEC).
- Phép nhân bình thường như DEC, nhưng phép cộng có khác: $\mathbf{1_2 + 1_2 = 10_2}$**.**

- **Hệ bát phân (OCTal)** gồm 8 chữ số 0, 1, … và 7. (Xem thêm trên mạng). Công thức: $\mathbf{(10_8)^n = 8^n}$**,** ở hệ DEC**.** Hệ này ít dùng!

- **Hệ thập lục phân (HEXadecimal)** gồm 16 chữ số 0, 1, …, 9, A, B, C, D, E

và F. Có thể dùng chữ in thường/HOA đều được.

- Đếm ở hệ HEX như sau: 0, 1, …,9, A, …, F, 10, 11, ..., FF, 100, 101, …, FFF, 1000,… Công thức**: $(10_{16})^n = 16^n$**, ở hệ DEC**.**

Chú ý:

Để ám chỉ hệ HEX ta viết thêm chỉ số dưới là 16 (trong Toán học), chữ $ (trong ngôn ngữ Pascal), 0x hay 0X (trong ngôn ngữ C/C++), # (trong HTML) phía trước chữ số đầu tiên hoặc chữ h (trong Assembly) phía sau.

Chẳng hạn: $1f_{16}$, $1F_{16}$, $1F, #1F, 1Fh, #1f, $1f, 0x1F, 0X1f là như nhau.

b)-Chuyển đổi một số từ các hệ đếm sang DEC

- Phương pháp dễ nhớ để đổi một số các hệ BIN, OCT hay HEX ra DEC là dùng lược đồ **Horner.** Để tính giá trị của đa thức $P(x) = ax^3 + bx^2 + cx + d$, ta không tính (cơ bắp) băng cách tính các đơn thức rồi cộng lại, mà theo cách của Horner là $P(x) = ((ax + b)x + c)x + d$.

	a	b	c	d
Đối số = x	P = a	P = xP + b	P = xP + c	P = xP+d

Ví dụ 1:

1101_2 = ? (DEC)

	1	1	0	1
Đối số = 2	1	3	6	13 (kết quả)

Ví dụ 2:

$2A3C = ? (DEC)

	2	10	3	12
Đối số = 16	2	42	675	10812 (kết quả)

Chuyển đổi số tự nhiên từ hệ DEC ra các hệ đếm khác:

Theo Horner ta có: $P = ((ax + b)x + c)x + d$.

P ban đầu chia cho x được $((ax + b)x + c)$ dư d. Đặt P mới là $(ax + b)x + c$.

P mới chia cho x được $(ax + b)$ dư c. Đặt P mới là $ax + b$.

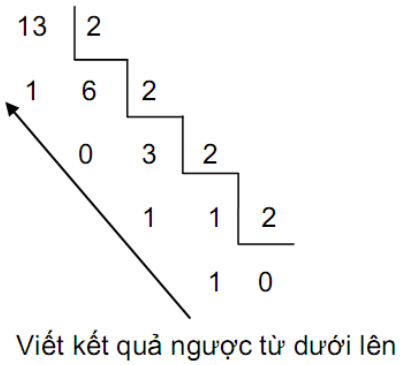
P mới chia cho x được a dư b. Đặt P mới là a.

P mới chia cho x được 0 dư a. (Để ý rằng a, b, c, d đều phải bé hơn x).

Như vậy, ta lần lượt thu được d, c, b và a.

Ví dụ 1:

13 = $\overline{abcd}_2$ (BIN).



Tức là tính các bit a, b, c và d?
Ta có:
13 chia cho 2 được 6 dư 1.
6 chia cho 2 được 3 dư 0.
3 chia cho 2 được 1 dư 1.
1 chia cho 2 được 0 dư 1.
Kết quả là 1101_2. (Viết từ dưới lên!).

Ví dụ 2:
213 = ? (HEX). Tức là tính các bit a, b, c và d?
Ta có:
213 chia cho 16 được 13 dư 5, (chữ số 5 trong hệ HEX).
13 chia cho 16 được 0 dư 13, (chữ số D trong hệ HEX).
Kết quả là $D5_{16}$. (Viết từ dưới lên!).

Chú ý:
- Chuyển đổi giữa HEX và BIN có thể làm như sau: Coi mỗi chữ số của hệ HEX như dãy 4 bít,

$0 = 0000_2$, $1 = 0001_2$, $2 = 0010_2$, $3 = 0011_2$, $4 = 0100_2$, $5 = 0101_2$, $6 = 0110_2$, $7 = 0111_2$, $8 = 1000_2$, $9 = 1001_2$, $A = 1010_2$, $B = 1011_2$, $C = 1100_2$, $D = 1101_2$, $E = 1110_2$, $F = 1111_2$,

thì việc chuyển đổi này dễ dàng thực hiện được.
Ngoài ra bên MTĐT có thể chuyển đổi và thực hiện một số phép tính ở các hệ đếm nêu trên và chuyển đổi giá trị qua các hệ đếm nhờ phần mềm **Calculator,** chọn Scientific ở menu **View.** Qua View, ta còn có thể hiện các số theo nhiều cách khác và có thể Copy để Paste vào nơi khác hoặc ngược lại.

2.3.2. Biểu diễn thông tin (mã hóa thông tin và giải mã dữ liệu)

1.- Biểu diễn dữ liệu số

- **Số tự nhiên nhỏ (trong Pascal gọi là kiểu Byte, trong C/C++ gọi là unsigned char)** thành dạng 8 bit (1 Byte) từ 00000000 đến 11111111 tức là giá trị là 0 đến 255. Khi vượt quá 8 bít thì bít thứ 9 bị bỏ đi, chẳng hạn: **255 + 1 = 0** . Muốn chính xác phải để kết quả ở kiểu lớn hơn, chẳn han dạng 2B dưới đây.

- **Số tự nhiên (trong Pascal gọi là kiểu Word, trong C/C++ gọi là unsigned int)** thành dạng 16 bit (2 Bytes) từ 0000000000000000 đến 1111111111111111, tức là giá trị là 0 đến 65535. Khi vượt quá 16 bít thì bít thứ 17 bị bỏ đi, Chẳng

hạn: **65535 + 1 = 0** . Muốn chính xác phải để kết quả ở kiểu cỡ lớn hơn.

- **Số nguyên ngắn (trong Pascal gọi là kiểu ShortInt, trong C/C+ gọi là char)** thành dạng 8 bit (1 Byte): Bit đầu là bít dấu (0 là +, 1 là -), 7 bít sau là các bít số. Giá trị số kiểu này từ 10000000 đến 01111111 tức là giá trị là -128 đến 127. Khi vượt quá 127 thì quay về -128, chẳng hạn: $01111111_2 + 1_2 = 10000000_2$,
tức 127 + 1 = -128. Muốn chính xác phải để kết quả ở kiểu lớn hơn.

Công thức mã hóa số nguyên:

Nếu Z ≥ 0 thì chỉ cần đổi sang hệ BIN;

Trái lại, dùng phương pháp "bù 2": **-Z = not(Z) + 1.**

Ví dụ: Mã hóa số -13 ta có -13 = not(13) + 1, trong hệ BIN. Cụ thể là:

$13 = 00001101_2$,

$not(13) = 11110010_2$, (Đảo bít $0 \leftrightarrow 1$) //bù 1

$not(13) + 1 = 11110011_2$ là mã của -13, //bù 1.

- **Số nguyên (trong Pascal gọi là kiểu Integer, trong C/C++ gọi là int)** thành dạng 16 bit (2 Bytes): Bit đầu là bít dấu (0 là +, 1 là -), 15 bít sau là các bít dấu. Giá trị từ số kiểu này là từ -32768 đến 32767. Khi vượt quá 32767 thì về -32768, chẳng hạn: 32767 + 1 = -32768 . Muốn chính xác phải để kết quả ở kiểu lớn hơn.

Công thức mã hóa số âm như số nguyên ngắn, tuy nhiên, mã phải có 16 bít.

Ví dụ: Mã hóa số -13 ta có -13 = not(13) + 1, trong hệ BIN. Cụ thể là:

$13 = 0000000000001101_2$,

$not(13) = 1111111111110010_2$, (Đảo bít $0 \leftrightarrow 1$) //bù 1

$not(13) + 1 = 1111111111110011_2$ là mã của -13 với cỡ 16 bít, //bù 1.

- **Số nguyên dài (Pascal gọi là số kiểu LongInt, trong C/C++ gọi là long int)** thành dạng 32 bit (4 Bytes): Bit đầu là bít dấu (0 là +, 1 là -), 31 bít sau là các bít số. Giá trị từ -2147483648 đến 2147483647.

Trong khi làm toán số vượt quá 2147483647 thì quay về -2147483648, chẳng hạn: 2147483647 + 1 = -2147483648 . Muốn chính xác phải để ở kiểu lớn hơn: Ở Pascal sang số thực, ở C/C++ sang số nguyên lớn. (C mạnh hơn Pascal).

- **Số tự nhiên lớn (trong Pascal không có, trong C/C++ gọi là unsigned long)** thành dạng 32 bit từ
$00000000000000000000000000000000_2$ đến
$11111111111111111111111111111111_2$,
tức là giá trị là 0 đến $2^{32} - 1 = 4294967295$. Khi làm toán vượt quá 32 bít, bít thứ 33 bị bỏ đi, chẳng hạn: 4294967295 + 1 = 0.
Muốn chính xác phải để kết quả ở kiểu số thực.

- **Số thực (Pascal gọi là kiểu Real, C/C++ gọi là float)** viết thành số thập phân **dấu phẩy động (dạng khoa học**, ví dụ: $3.17*10^3$ hoặc $4.3*10^{-3}$) hay dạng **dấu phẩy tĩnh (dạng thông thường**, ví dụ: 345.7645).

- Cách mã hóa số thực là viết dưới dạng khoa học chuẩn là gồm: Một số thập phân trong khoảng (-1;1) và nhân với 10 mũ một số nguyên, chẳng hạn: 3141592 viết thành $0.3141592*10^8$.
- Do vậy muốn mã hóa một số thực chỉ cần mã hóa cặp số số nguyên : Phần định trị và Phần đặc tính.

Chú ý:
- Dạng thông tin nào biểu diễn được qua các số thì mã hóa được, rồi sau lại giải mã được.
- Thông tin nào chưa có công cụ kỹ thuật biểu diễn qua số được thì tạm thời chịu, ví dụ mùi/vị. (Thực ra họ đã làm được nhưng chưa phổ biến ra mà thôi!)

Ví dụ:
- Số phức z = a + bi, a và b là số thực, vậy cũng sẽ mã hóa được qua 4 số nguyên.
- Các nốt nhạc biểu diễn được qua tần số (số tự nhiên) và thời gian (đo bằng mili giây, cũng là số tự nhiên), nên dễ mã hóa chúng!
- Mùi vị hiện nay chưa được công bố cách mã hóa/giải mã nó.

Chú ý:
- Khái niệm mã hóa rộng hơn không nhất thiết phải là kỹ thuật số hóa, ví dụ mã hóa ngành học, thẻ sinh viên, thẻ thư viện, biển kiểm soát xe cộ, chuyến bay, ...
- Còn **số hóa là mã hóa thành dữ liệu cho máy tính**, tức là thành dạng dữ liệu dãy bít, để đưa vào máy tính xử lý được ngay!
- Thời đại CNTT hiện nay là **thời đại số hóa, kỷ nguyên số hóa, kỷ nguyên kỹ thuật số**! Hơn thế nữa còn là thởi đại **Trí tuệ nhân tạo 4.0.**

2.- Kí tự và xâu kí tự

- **Bảng mã ASCII (American Standard Codes for Information Interchange)** gồm 256 kí tự: Mỗi kí tự có số thứ tự từ 0 đến 255 (Mã thập phân) và từ 00000000_2 đến 11111111_2 (Mã nhị phân) của nó. Không có kí tự trống!
- **Xâu (Chuỗi) kí tự (String)** là ghép các kí tự lại với nhau. Thậm chí không có kí tự nào, gọi là xâu trống, hay xâu rỗng.

Ví dụ:

"TIN" ↔ 84-73-78 ↔ 01010100-01001001-01001110.
"2010" ↔ 50-48-49-48 ↔ 00110010-00110000-00110001-00110000.

Bảng mã Unicode gồm 65536 kí tự đếm từ 0 đến 65535 (gọi là mã thập phân) nên dùng 16 bit (2 Byte) để mã hóa kí tự (gọi là mã nhị phân) của nó.

3.- Mầu sắc và hình ảnh

a)- Mầu cơ bản

- Ban đầu máy tính trong hệ điều hành DOS, giao diện văn bản, có thể cho 16 mầu cơ bản, do đó người ta dùng hệ đếm HEX để mã hóa hay hơn so với DEC.

$0 = 0 Đen (Black)	**$8 = 8 Xám (DarkGray)**
$1 = 1 Xanh tối (Blue)	**$9 = 9 Xanh sáng (LightBlue)**
$2 = 2 Lá tối (Green)	**$A = 10 Lá sáng (LightGreen)**
$3 = 3 Trời tối (Cyan)	**$B = 11 Trời sáng (LightCyan)**
$4 = 4 Đỏ tối (Red)	**$C = 12 Đỏ sáng (LightRed)**
$5 = 5 Tím tối (Magenta)	**$D = 13 Tím sáng (LightMagenta)**
$6 = 6 Nâu (Brown)	**$E = 14 Vàng (Yellow)**
$7 = 7 Ghi (LightGray)	**$F = 15 Trắng (White).**

- Trong **hệ điều hành DOS (Disk Operating System)**, màn hình ở chế độ văn bản, thường là 80 cột x 25 dòng, giao của mỗi cột và mỗi dòng là một kí tự. Mỗi kí tự có các thành phần: 1 Byte **mã của kí tự (code ASCII)** và 1 Byte **thuộc tính (Attribut)** của kí tự (mầu nền, mầu chữ và cả sự nhấp nháy hay không).
- Thuộc tính hay viết ở hệ HEX bằng số XY, với quy ước:
- Ở chế đọ thuần DOS, n**ếu X** > 7 thì mầu nền = X – 8 và chữ thì nhấp nháy, còn Y là mầu chữ. Còn DOS ở HĐH Windows thì có thể dùng mầu nền luôn là X.

Ví dụ 1:
Thuộc tính $4E = 01001110_2 là (không nhấp nháy (0)), nền đỏ (4 bít đầu là 0100_2), chữ vàng (4 bít sau là 1110_2).
Kí tự 'A' nền xanh chữ trắng không nấp nháy có mã = 65, thuộc tính = $1F

- Sau này ở **Hệ điều hành Windows**, giao của cột và dòng là một **Điểm ảnh (Pixel)** nó mang thông tin về mầu là một dãy n bit. Số lượng mầu sẽ bằng 2^n. Thế thì ta có thể tính được cỡ của một hình ảnh khi biết số điểm ảnh của nó!

B)- Hình ảnh (Image)

- Là một tập hợp các điểm ảnh, mỗi điểm mang thông tin về một mầu.
- Cách khác để mã hóa là dùng phương pháp véc tơ cho đỡ tốn bộ nhớ, không cần cho thông tin từng điểm.

Ví dụ:
Đường tròn có tâm tại điểm ảnh ở tọa độ (120,50), bán kính r = 100 với mầu đỏ, thì ta mã hóa nó thành bộ số (120,50,100,12). Khái niệm véc tơ ở đây có thể hiểu như là dãy hữu hạn các thành phần.

- Hiện nay chất lượng mầu ít nhất là 16 bit tức là có 2^{16} hay 65536 mầu. Mầu có mã thập phân từ 0 đến 65535, tức là ở BIN 0000000000000000_2 đến 1111111111111111_2. Cao hơn nữa là 24 bit và 32 bít, ...
- Trong Internet, người ta dúng loại 24 bít, dùng mẫu RGB để mã hóa mầu: **RGB (Red, Green, Blue),** tức là #RRGGBB, ở đây R, G, B là các chữ số của hệ HEX. #RR là độ đậm của mầu đỏ (Red), #GG của mầu lá (Green) và #BB của mầu xanh (Blue), so với độ đậm nhất là #FF.

Ví dụ 1:
#102A1F là mã hóa của mầu trộn của độ đỏ = #10, nghĩa là #10/#FF = 16/255 = 6.27%, độ lá là #2A/#FF = ..., độ xanh là #1F/#FF = ... (Bạn thử tính nhé). #FF0000 là mầu đỏ 100%, #0000FF là blue 100%, #FF00FF là trộn mầu đỏ với mầu xanh cho mầu tím 100%, #000000 là mầu đen, #FFFFFF là mầu trắng.

Ví dụ 2:
Mã nguồn Code của trang web: Viết "Chao cac ban" có cỡ là 7, font kiểu Arial và mầu là tím.

```
<html>
<body>
<font face=arial size=7 color=#FF00FF> Chao cac
ban! </font>
</body>
</html>
```

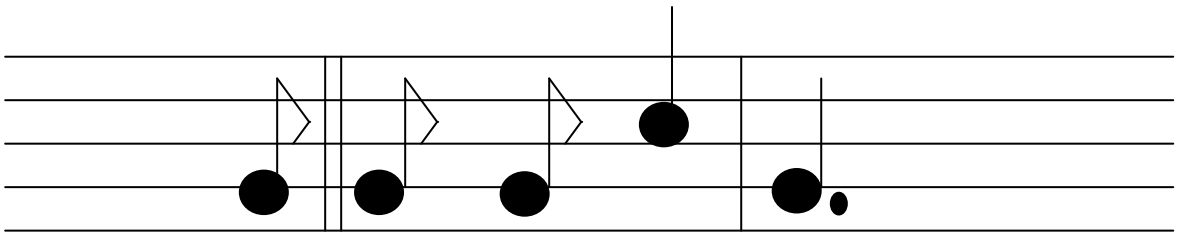
- Bạn hãy thay mã mầu trên bằng một số khác, lưu mã nguồn lại (Save As) dưới một tên nào đó và đuôi là .htm, chẳng hạn **test.htm** và Open nó bằng một trình duyệt quen biết như Chrome, Internet Explorer, CocCoc sẽ thấy cụ thể!
- Ngược lại, thấy mầu mà muốn biết mã thì cần dùng phần mềm chuyên dụng như Photoshop hay tra trên trang web:

4.- Âm thanh

- Dao động của vật chất gây ra âm thanh. Số lần dao động trong một giây gọi là tần số, đơn vị là Hertz hay viết tắt là Hz. Tai người nghe được các tần số từ 16 Hz đến 22.000 Hz. Âm thanh có tần số < 16 Hz gọi là hạ âm, cao hơn 22.000 Hz gọi là siêu âm. Có một vài loài vật nghe được Hạ âm/siêu âm.
- Mỗi âm thanh còn được xác định bởi thời gian nó phát ra.
- Tấn số, thời gian đều là các số nên mã hóa được thành cặp (Tần số, Thời gian).
- Một âm thanh phức tạp như bản nhạc chẳng hạn là một chuỗi các âm thanh.
- Âm thanh ngày càng được mã hóa tốt hơn… Ở DOS tần số và thời gian là các số tự nhiên. Ở Windows hệ thống âm thanh được xử lý tốt hơn nhiều. Có nhiều định dạng khác nhau, như WAV, MP3,.. Mỗi dạng lại có chất lượng khác nhau, chẳng hạn MP3 128 kbps, …, hay 320 kbps. Chất lượng 128 kbps chí cho âm thanh đến tần số từ 16 đến 17.000Hz, còn chất lượng 320kbps cho âm thanh tới 20.000 hz nghe được đủ dải tần… Nếu chuyển đỏi từ 128kbps sang 320kbps thì âm có thể mịn hơn nhưng tần số không được nâng lên!

Ví dụ:

Sòl, sòl, sòl, đô, sòl,… → (384;200), (384;200), (384;200), (512;400), (384;600),…



2.3.3. Các phép toán logic và mạch điện tử

- Mệnh đề toán học là mệnh đề mà chỉ có thể nói là đúng hoặc sai.
- Cũng như trong đại số: Một mệnh đề phụ thuộc vào một hay vài biến mệnh đề gọi là một hàm mệnh đề, nó trả lại giá trị Đúng hoặc Sai, chẳng hạn:

  Hàm một biến: f(x) = (x > 3),

  Hàm hai biến: f(x,y) = (x < y).
- Giá trị của một mệnh đề toán học được mã hóa là 1 (**đúng**) và 0 (**sai**).
- Cho 2 mệnh đề α và β ta có thể tạo ra các mệnh đề toán học mới bằng cách nối chúng lại, chẳng hạn: α **AND** β, α **OR** β, α **XOR** β, **IF** α **THEN** β, **NOT**(α).
- Ta sẽ quan tâm tới các dạng mệnh đề phức hợp đó! Chúng sẽ được dùng tới trong Excel hay lập trình chẳng hạn ở C/C++, Pascal, …

Ví dụ:

Mệnh đề "Ai đi đâu đấy, hỡi ai?", "Ôi, Hà Nội!", "Anh Minh ơi !" chỉ là những mệnh đề văn học.
Còn (Mặt trăng sáng hơn mặt trời), (3 > 2), … là những mệnh đề toán học.
Mệnh đề (x > 3) tuy chưa biết đúng/sai, nhưng khi thay x cụ thể vào sẽ biết được.

Sau đây là các phép toán mệnh đề toán học cơ bản :

1.- Phép hội

Cho hai mệnh đề a, b thì mệnh đề **(α và β)** gọi là **Hội của α, β**.
Mệnh đề hội chỉ đúng khi cả hai cùng đúng.
Hội (a và b) cũng có thể viết **(α and β), (α ∧ β), α.β, αβ** hay dưới dạng hàm mệnh đề **and(α,β)** như trong **Excel, (α)&&(β)** trong **C/C++.**

Ví dụ:
(6 chia hết cho 2) ∧ (6 chia hết cho 5) là sai, vì có một mệnh đề sai.

2.- Phép tuyển

Cho hai mệnh đề a, b thì mệnh đề **(α hoặc β)** gọi là **tuyển của α, β.**
Mệnh đề tuyển chỉ sai khi cả hai cùng sai.
Tuyển (a hoặc b) cũng có thể viết **(α or β), (α ∨ β), α + β** hay dưới dạng hàm mệnh đề **or(α,β)** trong **Excel, (α)||(β)** trong **C/C++.**

Ví dụ:
(6 chia hết cho 4) ∨ (6 chia hết cho 3) là đúng, vì có một mệnh đề đúng.

3.- Phép kéo theo

Cho hai mệnh đề a, b thì mệnh đề **(nếu α thì β)** gọi là **phép kéo theo của α, β**.
Mệnh đề kéo theo chỉ sai khi α đúng và β sai.
Kéo theo của a và b cũng có thể viết **(α suy ra β), (if α then β), α ⇒ β, α → β** hay dưới dạng biểu thức mệnh đề **α:β:γ** tức là **nếu α thì β, trái lại thì γ.**

Ví dụ:
Một anh nói: Tối nay tôi sẽ chuyển tiền cho anh. Nhưng chờ đến tận qua đêm rồi mà không thấy tiền chuyển, vậy thì anh kia đã sai.
Phép suy luận: $(4 = 4) \Rightarrow ((-2)^2 = 2^2) \Rightarrow (-2 = 2) \Rightarrow (4 = 0) \Rightarrow (1 = 0)$, là sai.

Nhà toán học Fermat nói: (Nếu n là số tự nhiên thì $2^{2^n} + 1$ là một số nguyên tố).
Về sau, người ta phát hiện ra giả thuyết đó là sai, chẳng hạn:
Với n = 5 thì số đó bằng $2^{32} = 4294967297 = 641 \times 6700417$ là hợp số.

4.- Phép phủ định

Cho mệnh **Không α** gọi là **phủ định của α**. Nó **đối kháng** với a, nghĩa là trái ngược giá trị của nhau. "Không α" cũng có thể viết ở dạng hàm **not(α), α' ,** $\neg\alpha$ hoặc $\overline{\alpha}$ hay **α!**

Chỳ ý:

Đối kháng khác với bất đồng, chẳng hạn: α = 'cái bảng này mầu đen', β='Cái bảng này mầu trắng', γ='Không phải cái bảng này mầu đen'. Khi đó: α, γ là đối kháng, α, β là bất đồng, còn β, γ cũng là bất đồng.

5.- Phép hoặc loại trừ

Cho hai mệnh đề a, b thì mệnh đề (**Chọn một trong hai)** gọi là **hoặc loại trừ của α, β**.

Mệnh đề Hoặc loại trừ sai khi và chỉ khi cả hai cùng đúng hoặc cùng sai.

Hoặc loại trừ của a,b cũng có thể viết **α xor β** hoặc $\alpha \veebar \beta$ hay dưới dạng hàm mệnh đề **xor(α,β) trong Excel.**

- Từ những phép tính trên ta có thể tạo ra những biểu thức phức tạp hơn.
- Tuy nhiên mỗi biểu thức logic đều có thể biểu diễn qua 3 phép toán là AND, OR và NOT, chẳng hạn (yêu cầu thuộc):

$$\alpha \rightarrow \beta = \neg\alpha \vee \beta = \overline{\alpha} + \beta$$

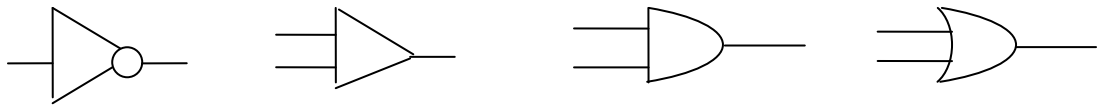
$$\alpha \text{ XOR } \beta = \alpha \veebar \beta = \overline{\alpha}\,\beta + \alpha\,\overline{\beta}$$

6.- Mạch điện tử

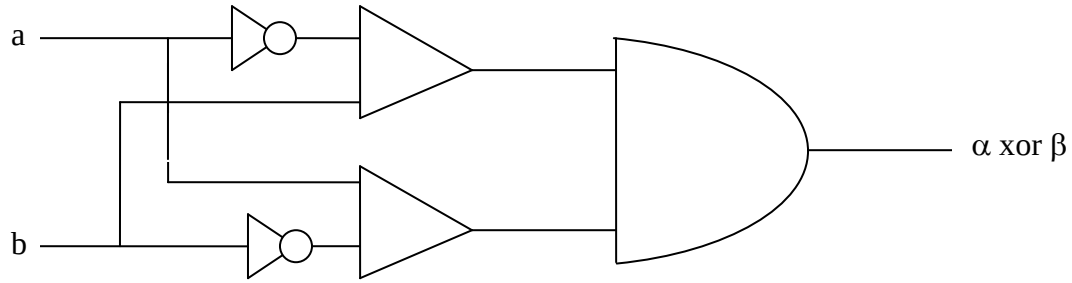
a)- Khái niệm

- Ở các nước tiên tiến người ta tạo ra được những con chip dựa trên kiến thức vi mạch logic. Trên 1cm^2 silic, người ta cấy ghép hàng ngàn vi mạch để tạo ra con chip lớn hơn có nhiều chức năng phức. Việt Nam chưa có công nghệ này mà chỉ tạm dừng lại ở chỗ mua các con chíp về lắp ráp thành các thiết bị điện tử và tin học thôi.
- Phép XOR rất hay dùng nên người ta cũng liệt nó vào hàng cơ bản!
- Quy ước:

Các mạch NOT, AND, OR và XOR lần lượt được vẽ như sau (đầu vào từ bên trái, đầu ra bên phải).



Sơ đồ nguyên lý chi tiết của mạch XOR như sau:



- Biết bảng giá trị của một biểu thức mệnh đề, ta có thể tìm ra biểu thức mệnh đề cụ thể đó. Ta dựa vào các kết quả đúng của biểu thức mà suy ra biểu thức tường minh, rồi rút gọn đi trước khi thiết kế mạch điện tử.
- Tin học kết hợp với Điện tử thì có thể làm ra được rất nhiều thiết bị điện tử!.

Ví dụ

Để lắp một bóng đèn cho cầu thang: Đèn đang tối, đến chân cầu thang ta bật công tắc A thì đèn sáng, lên trên gác ta bật công tắc B thì đèn tắt. Rõ ràng mạch này là A xor B.

b)- Vài tính chất quan trọng của các phép toán

- **Tính lặp** của phép hội và phép tuyển:

  $\alpha.\alpha.\ldots\alpha = \alpha$, $\alpha + \alpha + \ldots + \alpha = \alpha$.
- Hằng đẳng thức với mọi mệnh đề α:

  $\alpha.1 = \alpha$, $\alpha + 0 = \alpha$, $\alpha.0 = 0$, $\alpha + 1 = 1$, $\alpha.\overline{\alpha} = 0$, $\alpha + \overline{\alpha} = 1$.
- Phép hội và phép tuyển có tính chất **Giao hoán, Kết hợp** như số học:

  $\alpha.\beta = \beta.\alpha$, $\alpha.(\beta.\gamma) = (\alpha.\beta).\gamma$, $\alpha + \beta = \beta + \alpha$, $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$.
- Hai phép hội và tuyển có tính chất phân phối như số học:

  $\alpha.(\beta + \gamma) = \alpha.\beta + \alpha.\gamma$, và đặc biệt $\alpha + (\beta.\gamma) = (\alpha + \beta).(\alpha + \gamma)$.
- **Luật Đờ Moocgan**:

  $\overline{\alpha.\beta} = \overline{\alpha} + \overline{\beta}$, $\overline{\alpha + \beta} = \overline{\alpha} . \overline{\beta}$.

Ví dụ:

- (Một người đi bán kiếm và bán lá chắn. Anh ta rao: Hãy mua kiếm của tôi đi. Kiếm của tôi đâm cái gì cũng thủng. Và rao: Hãy mua lá chắn của tôi đi, không cái gì đâm thủng được)
- Không phải: ((5 chia hết cho 3) và (6 chia hết cho 3)) = (5 không chia hết cho 3) hoặc (6 không chia hết cho 3).
- Không phải: ((5 chia hết cho 3) hoặc (10 chia hết cho 3)) = (5 không chia hết cho 3) và (10 không chia hết cho 3).

c)- Một số ví dụ và bài tập điển hình khác về mạch điện tử

Ví dụ 1.

Trong một cuộc điều tra có 3 nhân chứng A, B và C cùng ngồi với nhau và nghe ý kiến của từng người. Cuối cùng chủ tọa hỏi lại để tìm xem tin ai, thì: A và B đối lập nhau, B và C đối lập nhau và C thì bảo A và B đều nói sai cả. Vậy ban điều tra tin ai?

Giải:

Theo đầu bài ta có các đẳng thức mệnh đề sau:

$A\overline{B} + \overline{A}B = 1$ (1), đọc là (A đúng và B sai) hoặc (A sai và B đúng) là mệnh đề đúng.

$B\overline{C} + \overline{B}C = 1$ (2), đọc là (B đúng và C sai) hoặc (B sai và C đúng) là mệnh đề đúng.

Nhân (làm phép và) 2 đẳng thức, vế với vế, ta thu được:

$(A\overline{B} + \overline{A}B).(B\overline{C} + \overline{B}C) = 1$

$A\overline{B}B\overline{C} + A\overline{B}\,\overline{B}C + \overline{A}B.B\overline{C} + \overline{A}B\overline{B}C = 1.$

Số hạng đầu và cuối bằng 0 (tức sai) vì có B và $\overline{B}$ phủ định lẫn nhau trong phép và, đồng thời BB vẫn bằng B, $\overline{B}\,\overline{B}$ vẫn bằng $\overline{B}$ nên chỉ còn lại:

$A\overline{B}C + \overline{A}B\overline{C} = 1.$ (3).

Theo bài ra ta còn có: $C\overline{A}\,\overline{B} + \overline{C}A + \overline{C}B = 1.$ (4).

Nhân (3) và (4) lại và rút gọn ta được $\overline{A}B\overline{C} = 1$, tức là A sai, B đúng và C sai. Tóm lại B là nhân chứng nói đúng!

Ví dụ 2.

Ở 2 bên đường có 2 làng A và B. Dân làng A thi luôn nói thật, hỏi 1 điều đúng thì gật đầu, sai thì lắc đầu. Dân làng B thi luôn nói dối, hỏi 1 điều đúng thì lắc đầu, sai thì gật đầu. Một người khách lạ đến 1 trong 2 làng đó, nhưng không biết mình đang ở làng nào, gặp 1 người dân, không biết dân làng nào, vì họ đi qua lại giữa 2 làng. Người khách muốn hỏi đúng 1 câu để người dân cứ gật đầu thì biết mình đang ở làng A, lắc thì biết mình đang ở làng B. Bạn hãy giúp người khách này với!

Giải:

Gọi X là câu hỏi: "Đây là làng A?", Y là câu hỏi: "You là người làng A à?". Câu hỏi của người khách là một biểu thức tạo ra từ X và từ Y, và có bảng giá trị :

X Y Câu hỏi

1	1	1	(1)
1	0	0	(2)
0	1	0	(3)
0	0	1	(4)

Ý nghĩa

(1): Nếu đang đứng ở làng A và gặp người dân làng A thì muốn người đó gật, ta phải hỏi câu đúng.
(2): Nếu đang đứng ở làng A và gặp người dân làng B thì muốn người đó gật, ta phải hỏi câu sai.
(3): Nếu đang đứng ở làng B và gặp người dân làng A thì muốn người đó lắc, ta phải hỏi câu sai.
(4): Nếu đang đứng ở làng B và gặp người dân làng B thì muốn người đó lắc, ta phải hỏi câu đúng.
Từ bảng giá trị trên tìm được biểu thức tường minh của Câu hỏi:

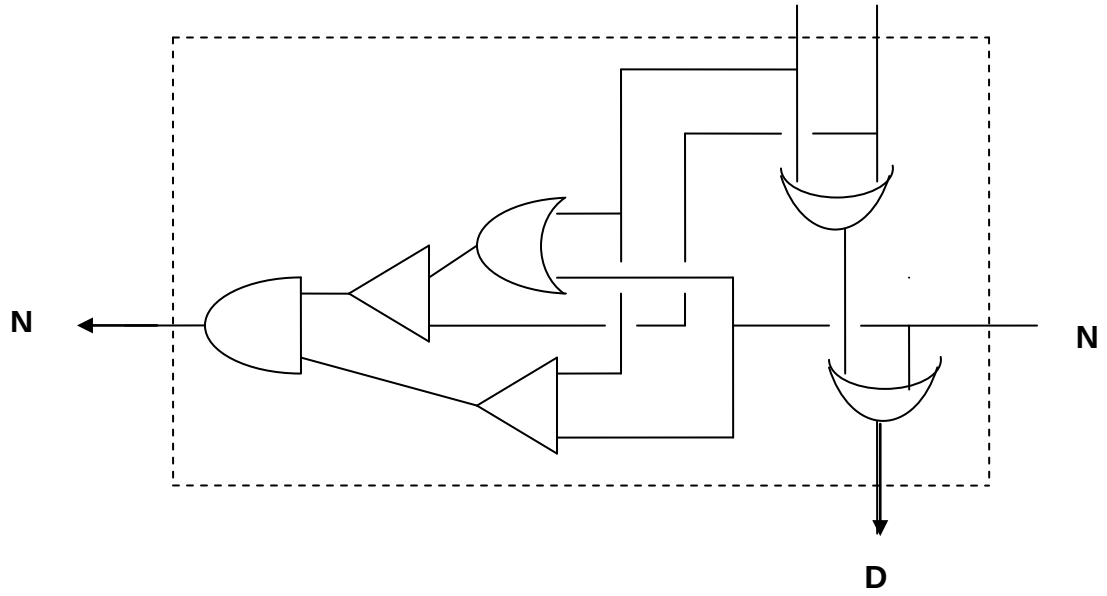
Câu hỏi = $XY + \overline{X}\ \overline{Y}$, tức là

"Đây là làng A à và ôu là người làng A à?" hoặc "Đây không là làng A à và You không là người làng A à?",
"Đây là làng A à và ôu là người làng A à?" hoặc "Đây là làng B à và You là người làng B à?",
Hay gọn hơn, ta có: Câu hỏi = "You là người làng này à?"

Sau đây ta xét một ví dụ rất tin học nữa của một linh kiện cho 2 đầu ra và ghép nối vào máy tinh ra sao: Chíp cộng nhị phân.
Như ta đã biết bảng cộng nhị phân: 0 + 0 = 0, 0 + 1 = 1 + 0 = 1 và 1 + 1 = 10.
Cũng giống như công các số ở các hệ đếm khác: "Cộng mấy với mấy được mấy nhớ mấy". Chi tiết hơn: "Cộng mấy với mấy và với nhớ cũ được mấy nhớ mấy".
Với hệ nhị phân ta có bảng mẫu sau đây:

A	**B**	**N**	**D**	**N**	Đọc là cộng A với B, với N cũ, được D và nhớ N mới.
0	0	0	0	0	
1	0	0	1	0	
0	1	0	1	0	
0	0	1	1	0	
1	1	0	0	1	
0	1	1	0	1	
1	0	1	0	1	
1	1	1	1	1	

A B



Từ đầu vào (A,B,N) và đầu ra là (D,N).

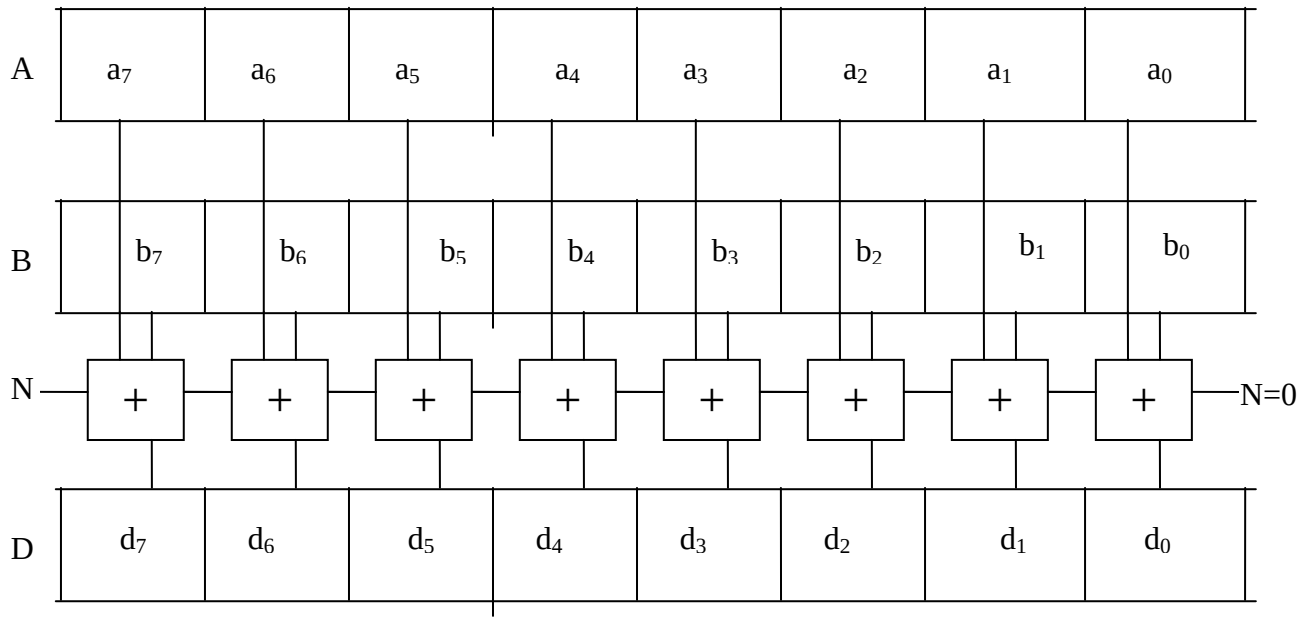
Dễ thấy: $D = A\overline{B}\ \overline{N} + \overline{A}\ B\ \overline{N} + \overline{A}\ \overline{B}\ N + ABN = \ldots = (A \veebar B) \veebar N,$

$N = AB\overline{N} + \overline{A}\ B\ N + A\overline{B}\ N + ABN = \ldots = (A \veebar N)B + AN,$

Chú ý là cách rút gọn giống như trên ví dụ 3.

Từ đó thiết kế được con chíp gọi là Bộ cộng 2 bít (A và B) cùng 1 bit nhớ (N) như hình vẽ bên trên, tạo ra một tế bào chíp Cộng.

Bây giờ ta hãy xem các bộ cộng 2 bít ghép nối với nhau thế nào để cộng 2 số tự nhiên cỡ 1 Byte, tức 8 bít, tức là số A + B được D nhờ dãy tế bào chíp Cộng như thế nào nhé:



Chú thích:

Hàng đầu là số A = $\overline{a_7a_6a_5a_4a_3a_2a_1a_0}$ (ở hệ nhị phân)

Hàng hai là số B = $\overline{b_7b_6b_5b_4b_3b_2b_1b_0}$ (ở hệ nhị phân, b_i là bít thư i từ bên phải)

Hàng ba là 8 bộ cộng 2 bít được ghép nối với nhau. Khi nhận được lệnh cộng A và B thì máy cộng lần lượt từ phải sang trái: Nhớ vào đầu là 0, nhớ ra của bộ cộng bên phải là nhớ vào của bộ cộng kế tiếp bên trái, bít được sẽ vào ô ở

Hàng thứ 4 là số D = $\overline{d_7d_6d_5d_4d_3d_2d_1d_0}$ (ở hệ nhị phân), kết quả của phép cộng.

Nếu trong chương trình khai báo A, B và kết quả D đều cùng kiểu số tự nhhiên cở 1 Byte, thì bit N ra cuối cùng dù bằng 0 hay 1 thì cũng không có chỗ để nữa.

2.3. Đôi điều về Thuật toán, Phần mềm và Hệ điều hành

- Việc xử lý thông tin mà con người phải nhờ đến MTĐT. Làm thế nào để MTĐT giải quyết đó là một **Bài toán Tin học.** Để giải nó, phải dạy cho máy cách làm theo một **Thuật toán (Algorithm)** nào đó càng hiệu quả càng tốt **.**
- Thuật toán phải gồm **hữu hạn** bước **khả thi**, **phổ dụng** và kết quả **đúng đắn**.
- Người ta viết thuật toán đó ra **Mã nguồn (Source Code)** bằng một **Ngôn ngữ lập trình (Programing Language)** rồi nhờ **Chương trình dịch (Compiler)** dịch ra thành **Mã máy (Binary Code)** để cho ra một **Chương trình (Program).** Một CT [cùng nhiều CT phụ họa và dữ liệu] để giải quyết một bài toán tạo ra một **Phần mềm (Soft Ware).** Phần mềm quan trọng nhất của MTĐT là **Hệ điều hành (HĐH).** Còn lại là phần mềm khác như ứng dụng, virus, giải trí,…
- Còn Phần cứng (**Hard Ware**) là các thiết bị lắp ráp thành một máy tính. Đặc biệt là các **Thiết bị Lưu trữ**, **Thiết bị Vào/Ra (Input/Output Devices).** Một đĩa cứng (**Hard Disk**) chia nhỏ thành nhiều **Phân vùng (Partition)** tạo ra các ổ đĩa cứng **C:, D:, E:\…** Mỗi ổ đĩa là **Thư mục gốc** của ổ đĩa, chẳng hạn **C:\. D:\,** chứa các **Folder** và các **Files**.
- Tất cả dữ liệu được tổ chức theo một **Cấu trúc hình cây** tùy theo mỗi Hệ điều hành (**Operating System**) cụ thể, có thể nói là có dạng **Gia phả.**
- **Phần cứng** và **Phần mềm** đua nhau phát triển với tốc độ như vũ bão!
- **Hệ điều hành** đã phát triển từ **DOS**, **WINDOWS, LINUX**, **UBUNTU**,...
- **USER** phải biết sử dụng máy tính của mình một cách khôn khéo, hợp lý, có lợi nhất cho mình. Bạn vào **http://khoia0.com** > mục **Sinh viên** để tự học thêm**!**

Phạm Đăng Long
Email: lightsmok@gmail.com,
ĐT: 0904070637, Zalo: (+84)986838536